



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/737,374	12/16/2003	Roger Hansen	200312027-1	5369
22879	7590	10/31/2007	EXAMINER	
HEWLETT PACKARD COMPANY			TRUONG, LOAN	
P O BOX 272400, 3404 E. HARMONY ROAD			ART UNIT	PAPER NUMBER
INTELLECTUAL PROPERTY ADMINISTRATION				
FORT COLLINS, CO 80527-2400			2114	
MAIL DATE		DELIVERY MODE		
10/31/2007		PAPER		

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

m/n

Office Action Summary	Application No.	Applicant(s)
	10/737,374	HANSEN ET AL.
	Examiner	Art Unit
	LOAN TRUONG	2114

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) Responsive to communication(s) filed on 13 June 2007.
- 2a) This action is **FINAL**. 2b) This action is non-final.
- 3) Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) Claim(s) 1-37 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) Claim(s) _____ is/are allowed.
- 6) Claim(s) 1-37 is/are rejected.
- 7) Claim(s) _____ is/are objected to.
- 8) Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) The specification is objected to by the Examiner.
- 10) The drawing(s) filed on _____ is/are: a) accepted or b) objected to by the Examiner.
 Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
 Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) All b) Some * c) None of:
 1. Certified copies of the priority documents have been received.
 2. Certified copies of the priority documents have been received in Application No. _____.
 3. Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413) |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | Paper No(s)/Mail Date. _____ |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO/SB/08) | 5) <input type="checkbox"/> Notice of Informal Patent Application |
| Paper No(s)/Mail Date _____. _____ | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. This office action is in response to the amendment filed June 12, 2007 in application 10/737,374.
2. Claims 1-37 are presented for examination. Claims 1, 3-7, 10, 12, 15-16, 21-22, 24, 26-28 and 30-37 are amended.

Response to Arguments

3. Applicant's arguments with respect to claims 1-37 have been considered but are moot in view of the new ground(s) of rejection.

In regard to claim 6 and claim 15 applicant disclosed, for claim 6, the limitation of "the persistent memory unit is configured to store multiple sets of checkpoint data sent from the processor at successive time intervals and the persistent memory unit provides the multiple sets of checkpoint data upon request by the backup process at one time." However, claim 6 only recite the limitation of "the persistent memory unit provides the multiple sets of checkpoint data through remote direct memory reads upon request by the backup process at one time." Claim 15, applicant disclosed the limitation of "providing multiple sets of checkpoint data to a backup process via a direct memory read command from the backup process." However, claim 15 only recite the limitation of "clearing the multiple sets of checkpoint data. It is unclear whether these limitations are meant to be incorporated in the above claims or reference elsewhere.

In regard to claim 30, applicant disclosed that claim 30 recites the limitation of "means for periodically supplying multiple sets of checkpoint data to the backup process in response to a request from the backup process" and the cited references do not teach or suggest providing

Art Unit: 2114

multiple sets of the checkpoint data but rather only one version at a time. However, claim 30 only recited "means for periodically accessing at least a portion of the multiple sets of checkpoint data in the backup process." Examiner interprets the multiple sets of checkpoint to equate to Chung et al. periodic/multiple snapshots of the running state of the primary application module where each snapshot is consider a set of data within a reference time period. For these reason, when the claim language is interpreted to its broadest interpretation, the rejection is maintained.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

The factual inquiries set forth in *Graham v. John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
 2. Ascertaining the differences between the prior art and the claims at issue.
 3. Resolving the level of ordinary skill in the pertinent art.
 4. Considering objective evidence present in the application indicating obviousness or nonobviousness.
-
4. Claims 1-5, 7-13, 15-16, 32-33 and 35-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chung et al. (US 6,195,760) in further view of Pandya (US 2004/0037319).

In regard to claim 1, Chung et al. teach a system for storing checkpoint state information, comprising:

a network interface to an external network (*network, fig. 1, 100, col. 3 lines 60-62*); and
a persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) coupled to the network interface (*connected to the network, fig. 1, 100, col. 4 lines 41-44*), wherein:

the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) is configured to receive the checkpoint data (*periodically receives from each fault-protected application module running on the network the most current state of that application, col. 4 lines 41-44*) via a memory write command from a primary process (*hot backup where each copies of an application can process client request and states are synchronized among multiple copies, col. 2 lines 7-14*), and to provide access to the checkpoint data via a memory read command from a backup process (*last operating state provided to the backup, col. 4 lines 34-40*), through the network interface (*last stored state is retrieved from the memory of Checkpoint Server connected to network, fig. 1, 110, 100, col. 4 lines 41-48*)); and

the backup process provides recovery capability in the event of a failure of the primary process (*idle or backup application module assume the functioning of a failed primary application module upon failure-detection, col. 4 lines 34-40*).

Chung et al. does not explicitly teach the system for storing checkpoint state information comprising of a remote direct memory read and write command and the persistent memory unit is configured with byte-level memory access granularity.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

It would have been obvious to modify the system of Chung et al. by adding Pandya TCP/IP processor and engine. A person of ordinary skill in the art at the time of applicant's invention would have been motivated to make the modification because it would provide a new high performance and low latency to relieve the host processor of the server performance impact (*paragraph 0067*).

In regard to claim 2, Chung et al. teach the system of claim 1, further comprising: a persistent memory manager (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) configured to provide address context information to the network interface (*pathname location of each copy of the application module on the host computer, fig. 2, 200*).

In regard to claim 3, Chung et al. teach the system of claim 1, wherein the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) is configured to provide access to the checkpoint data to another processor (*Checkpoint server transmit last stored state to new primary application module, fig. 1, 110, H1-6, col. 4 lines 41-48*), and the backup process is executed by the other processor (*backup for A application on multiple host computer, fig. 2, H2 and H3*).

Chung et al. does not explicitly teach providing a remote direct memory read access.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 4, Chung et al. teach the system of claim 1, wherein the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) provides the checkpoint data by the backup process after the primary process fails (*Checkpoint server transmit last stored state to new primary application module, fig. 1, 110, H1-6, col. 4 lines 41-48*).

Chung et al. does not explicitly teach providing data through remote direct memory read access.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 5, Chung et al. teach the system of claim 1, wherein the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) is configured to store multiple sets of checkpoint data sent from the processor at successive time intervals (*checkpoint technique to periodically take snapshots of the running state in a stable storage media, col. 1 lines 49-58*).

Chung et al. does not explicitly teach the system for storing data through remote direct memory writes.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 7, Chung et al. teach the system of claim 1, wherein the primary process (*primary for application A, fig. 2, 202*) writes the checkpoint data (*snapshot of the running state of the primary application, col. 1 lines 49-58*) to the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) independently from the backup process (*backup for application A, fig. 2, 203-205*).

Chung et al. does not explicitly teach the system for remote direct memory writes.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include

read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 8, Chung et al. teach the system of claim 1, wherein the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) is configured as part of a memory access-enabled system area network (*Checkpoint Server is connected to network, fig. 1, 110, 100*).

Chung et al. does not explicitly teach the system of claim 1, is configured as part of a remote direct memory access.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 9, Chung et al. teach the system of claim 1, wherein the persistent memory unit provide access information to authenticated remote processors (*application module register itself for its own failure and recovery process, col. 3 lines 1-15*).

Chung et al. does not explicitly teach the system where the persistent memory unit is configured with address protection and translation tables to authenticate requests from remote processors.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access which has the RDMA look-up table that include information like RDMA region and the access keys, and virtual address translation functionality (*paragraph 0124 lines 13-18*)

Refer to claim 1 for motivational statement.

In regard to claim 10, Chung et al. teach a method for recovering the operational state of a primary process, comprising:

a persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*);
receiving checkpoint data regarding the operational state of the primary process (*periodically receives from each fault-protected application module running on the network the most current state of that application, col. 4 lines 41-44*) in the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*); and
providing the checkpoint data to a backup process via a memory read command from the backup process (*last operating state of the failed application module must be provided to the backup application module, col. 4 lines 34-40*).

Chung et al. does not teach the method of mapping virtual addresses to physical address configured with byte-level memory access granularity and remote direct memory read and writes command.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 11, Chung et al. teach the method of claim 10, further comprising: providing context information regarding the addresses to the primary process and the backup process.

It is inherent that the network of fig. 1 with the plurality of host computer H1-H6 if connected in an Ethernet network would have their own IP address to distinct one host computer from another (*col. 3 lines 60-67*). Furthermore, the registration request from each failure-protected application module included a list of the host computers on which the application modules resided and where on each the executable program can be found (*col. 4 lines 49-60*).

In regard to claim 12, Chung et al. teach the method of claim 10, further comprising: providing the checkpoint data to the backup process upon failure of the primary process

(*Checkpoint server transmit last stored state to new primary application module, fig. 1, 110, H1-6, col. 4 lines 41-48*).

Chung et al. does not teach remote direct memory reads.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 13, Chung et al. teach the method of claim 10, further comprising: overwriting the checkpoint data with current checkpoint data (*when a failure of the primary application, the checkpoint data of the last stored state of the failed primary is supplied to the backup application module, col. 1 lines 49-57*).

In regard to claim 15, Chung et al. teach the method of claim 10, further comprising: clearing the portion of the multiple sets of checkpoint data (*periodically receives state updates from the primary application module, col. 1 lines 64-67*).

In regard to claim 16, Chung et al. teach the method of claim 10, further comprising:

allowing the backup process to read previously unread portions of the checkpoint data upon failure of the primary process (*Checkpoint server transmit last stored state to new primary application module, fig. 1, 110, H1-6, col. 4 lines 41-48*); and resuming functions performed by the primary process with the backup process (*checkpoint data of the last stored state of the failed primary application module is supplied to the backup application module, col. 1 lines 52-58*).

Chung et al. does not teach the method of remote direct memory read.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 32, Chung et al. teach a method for recording the operational state of a primary process, comprising: checkpoint data (*periodically receives from each fault-protected application module running on the network the most current state of that application, col. 4 lines 41-44*) regarding the operational state of the primary process (*WatchDog periodically polls application module and reports any failure to the ReplicaManager, col. 7 lines 42-49*) in a memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) via a memory access write command (*last stored state is retrieved from the memory of Checkpoint Server connected to network, fig. 1, 110, 100, col. 4 lines 41-48*).

Chung et al. does not explicitly teach the method for accessing checkpoint data in persistent memory via a remote direct memory access wherein the persistent memory unit is addressable at byte-level granularity.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 33, Chung et al. teach the method of claim 32, further comprising: overwriting the checkpoint data in the persistent memory unit with current checkpoint data via a direct memory access write command (*when a failure of the primary application, the checkpoint data of the last stored state of the failed primary is supplied to the backup application module, col. 1 lines 49-57*).

Chung et al. does not explicitly teach the method for transmitting data in persistent memory via a remote direct memory access.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 35, Chung et al. teach a method for retrieving the operational state of a primary process, comprising: transmitting a memory access read command via network to a memory unit from a backup process for the primary process (*the last stored state of that failed application module is retrieved from the memory of Checkpoint Server and provided to the new primary application module for continued processing, col. 4 lines 44-48*).

Chung et al. does not explicitly teach the method for transmitting a remote direct memory access read wherein the persistent memory unit is addressable at byte-level granularity.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 36, Chung et al. teach the method of claim 35, further comprising: periodically transmitting the memory access read command to retrieve at least a portion of the checkpoint data for the backup process (*Checkpoint Server periodically receives from each fault-protected application module running on the network the most current state of that application, col. 4 lines 41-44*).

Chung et al. and Stiffer et al. does not explicitly teach the method for transmitting the remote direct memory access.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 37, Chung et al. teach the method of claim 35, further comprising: transmitting the memory access read command to retrieve previously unread portions of the checkpoint data upon failure of the primary process (*Checkpoint server transmit last stored state to new primary application module, fig. 1, 110, H1-6, col. 4 lines 41-48*).

Chung et al. and Stiffer et al. does not explicitly teach the method for transmitting the remote direct memory access.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

5. Claim 6 is rejected under 35 U.S.C. 103(a) as being unpatentable over Chung et al. (US 6,195,760) in further view of Pandya (US 2004/0037319) in further view of Wang (US 7,082,553).

In regard to claim 6, Chung et al. does not explicitly teach the system of claim 5, wherein the persistent memory unit provides the multiple sets of checkpoint data through remote direct memory reads upon request by the backup process at one time.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

Chung et al. and Pandya does not teach the system wherein the persistent memory unit provides the multiple sets of checkpoint data upon request by the backup process at one time.

Wang teaches the for providing reliability and availability in a distributed component object model object oriented system by implementing checkpoint request to store current checkpoint of each object in registry image of disk storage (*fig. 8, col. 9 lines 21-30*).

It would have been obvious to modify the method of Chung et al. and Pandya by adding Wang distributed component object model object oriented system. A person of ordinary skill in the art at the time of applicant's invention would have been motivated to make the modification because it would help to quickly recover from a failure (*col. 9 lines 30-46*).

6. Claims 14 and 34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chung et al. (US 6,195,760) and in further view of Pandya (US 2004/0037319) in further view of St. Pierre et al. (US 6,141,773).

In regard to claim 14, Chung et al. and Pandya does not teach the method of claim 10, further comprising: appending updated checkpoint data to at least one previous set of the checkpoint data.

St. Pierre et al. disclosed the method of backing up and restoring data in a computer storage system where differential backup is formed by the identified changed segments omitting at least on the segments that has not been changed (*col. 5 lines 30-63*). A differential bit file may be constructed including copies of only the changed data segments (*fig. 13, 111a-111d*). The differential bit file captures changes to a logical entity as contiguous (*col. 17 lines 28-38*).

It would have been obvious to modify the method of Chung et al. and Pandya by adding St. Pierre et al. method of backing up data in a computer storage system (*col. 5*

lines 30-63). A person of ordinary skill in the art at the time of applicant's invention would have been motivated to make the modification because it would permits recovery from errors, including use of a mirror for data located at a remote facility, that also permits recoveries from catastrophic failure (*col. 5 lines 25-28*).

In regard to claim 34, Chung et al. and Pandya does not explicitly teach the method of claim 32, further comprising: appending updated checkpoint data to a previous set of the checkpoint data via a direct memory access write command.

St. Pierre et al. disclosed the method of backing up and restoring data in a computer storage system where differential backup is formed by the identified changed segments omitting at least on the segments that has not been changed (*col. 5 lines 30-63*). A differential bit file may be constructed including copies of only the changed data segments (*fig. 13, 111a-111d*). The differential bit file captures changes to a logical entity as contiguous (*col. 17 lines 28-38*).

Refer to claim 14 for motivational statement.

7. Claims 17-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chung et al. (US 6,195,760) in further view of Pandya (US 2004/0037319) in further view Traversat et al. (US 6,957,237).

In regard to claim 17, Chung et al. disclosed the method of claim 10, further comprising:

storing information (*Replica Manager's internal table, fig. 2, 200, col. 7 lines 45-49*) to the physical addresses of the checkpoint data (*ReplicaManager is made persistent by storing table in the Checkpoint Server, fig. 2, 110, 200; col. 8 lines 51-52*) in the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) when the primary process opens a memory region for the checkpoint data (*Checkpoint Server connected to network periodically receives the current state of that application, which state is stored in its memory, col. 4 lines 41-45*); and

Chung et al. and Pandya does not explicitly teach the method of storing access information to checkpoint data and providing the access information to subsequent requestors of the checkpoint data.

Traversat et al. teach the method of a database store for a virtual heap by implementing a migration of an application to another device with the lease information from the checkpointed state of the application to be used to re-established the leases (*col. 17 lines 4-11*), a leases is a grant of guaranteed access to a service (*col. 12 lines 45-62*).

It would have been obvious to modify the system of Chung et al. and Pandya by adding Traversat et al. virtual heap and paging. A person of ordinary skill in the art at the time of applicant's invention would have been motivated to make the modification because it would provide application migration and separate an application so that only it runs in a persistent heap (*col. 3 lines 28-41*).

In regard to claim 18 Chung et al. and Pandya does not explicitly disclosed the method of claim 17, further comprising:

establishing a connection to a process requesting access to the checkpoint data; and

binding the access information to the connection.

Traversat et al. teach the method of database stored for a virtual heap by implementing a Lease based service to grant a guaranteed access to a service over a period wherein the service may be external to the virtual machine or within which an application desiring a service is executing whether local or remote (*col. 12 lines 45-62*).

Refer to claim 10 for motivational statement.

In regard to claim 19, Chung et al. and Pandya does not teach the method of claim 17, further comprising:

verifying authentication information from the subsequent requestors.

Traversat et al. teach the method of database stored for a virtual heap by implementing a Lease based service granting access to many of the services in the Jini system environment (*col. 12 lines 45-62*).

Refer to claim 10 for motivational statement.

8. Claim 20 is rejected under 35 U.S.C. 103(a) as being unpatentable over Chung et al. (US 6,195,760) and in further view Pandya (US 2004/0037319) in further view of Ho et al. (US 2002/0073325).

In regard to claim 20, Chung et al. disclosed the method of claim 10, further comprising:

a persistent memory manager (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) during address protection and translation tables (*table, fig. 2, 200, col. 7 lines 1-5*) on the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*).

Chung et al and Pandya does not explicitly teach the method of authenticating a persistent memory manager during initialization.

Ho et al. teach the method of authenticating software licenses by implementing a persistent storage medium comprising a signature authentication program in the software protection program (*paragraph 0023*).

It would have been obvious to modify the system of Chung et al. and Pandya by adding Ho et al. method of authenticating software licenses. A person of ordinary skill in the art at the time of applicant's invention would have been motivated to make the modification because it would provide protection and allow legitimate backup copies (*paragraph 0011*).

9. Claims 21-23 and 25-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chung et al. (US 6,195,760) in further view of Stiffer et al. (US 6,622,263) in further view of Pandya (US 2004/0037319).

In regard to claim 21, Chung et al. disclosed a computer product, comprising: computer executable instructions (*application modules, col. 4 lines 1-15*) operable to:

allow memory access from a remote processor via a network (*Checkpoint Server connected to the network periodically receives the most current state of each fault-protected application module stored in its memory, col. 4 lines 40-45*), wherein the memory access references a persistent memory (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*);

Chung et al. does not teach a computer product reference to a persistent memory virtual address; store checkpoint data from a primary process; translate the virtual address to a physical address in the persistent memory unit wherein the persistent memory unit is addressable at byte-level granularity; and allow access to the checkpoint data for use in a backup process.

Stiffler et al. disclosed the apparatus for achieving system-directed checkpointing without specialized hardware assistance with the implementation of a memory map to convert virtual addresses used by application and by the operating system into physical addresses that point to specific locations in main memory. Each memory-map entry, in addition to containing virtual to physical address translation information, contains other information as well (*col. 5 lines 5-14*). Stiffler et al. also disclosed the method where each computer other than the standby sends checkpoint data to its neighbor on the right and each stores in its shadow memory (*col. 14 lines 40-46*), when a fault is detected, the computer to the right takes over the tasks that was executing prior to the fault (*col. 14 lines 47-55*).

It would have been obvious to modify the system of Chung et al. by adding Stiffler et al. apparatus for achieving system-directed checkpointing without specialized hardware assistance. A person of ordinary skill in the art at the time of applicant's

invention would have been motivated to make the modification because it would establishing and recording a consistent system state from which all running applications can be safely resumed following a fault (*col. 1 lines 13-17*).

Chung et al. and Stiffler et al. does not explicitly teach a computer product comprising a remote direct memory access.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 22, Chung et al. does not explicitly teach the computer product of claim 21, further comprising: computer executable instructions operable to: allow the processor to access address context information.

Stiffer et al. teach the method of checkpointing and fault recover software runs on standard platforms to detect observable malfunctions such as out-of-range address or exceeding a allocated range defined for a given data structure (*col. 4 lines 24-36*).

Refer to claim 21 for motivational statement.

In regard to claim 23, Chung et al. teach the computer product of claim 21, further comprising: computer executable instructions operable to:

store multiple updates to the checkpoint data sent at successive time intervals (*checkpoint technique to periodically take snapshots of the running state in a stable storage media, col. 1 lines 49-58*).

In regard to claim 25, Chung et al. disclosed the computer product of claim 21, wherein the persistent memory (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) is configured as part of a memory access-enabled system area network (*Checkpoint Server is connected to network, fig. 1, 110, 100*).

Chung et al. and Stiffer et al. does not explicitly teach the system of claim 21, is configured as part of a remote direct memory access.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 26, Chung et al. teach an apparatus comprising:
means for communicatively coupling (*configuring a fail-over process, col. 1 lines 36-37*)
a persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) to a network
(*network, fig. 1, 110, col. 4 lines 41-44*) that enables access to the memory unit (*A checkpoint*

Server connected to network periodically receives from each fault-protected application module running on the network, col. 4 lines 41-44);

means for receiving checkpoint data for a primary process (*periodically receives from each fault-protected application module running on the network the most current state of that application, col. 4 lines 41-44*) in the persistent memory unit (*Checkpoint Server, fig. 1, 110, col. 4 lines 41-44*) via the network (*network, fig. 1, 110, col. 4 lines 41-44*); and

means for allowing the primary process to access the checkpoint data (*last operating state of the failed application module must be provided to the backup application module, col. 4 lines 34-40*) via the network (*Checkpoint Server connected to network to receives from each fault-protected .*

Chung et al. does not explicitly teach the means for mapping virtual addresses of the persistent memory unit to physical addresses of the persistent memory unit.

Stiffler et al. disclosed the method of system-directed checkpointing without specialized hardware assistance with a memory map of virtual addresses to physical addresses (*col. 5 lines 2-8*).

It would have been obvious to modify the system of Chung et al. by adding Stiffler et al. apparatus for achieving system-directed checkpointing without specialized hardware assistance. A person of ordinary skill in the art at the time of applicant's invention would have been motivated to make the modification because it would establishing and recording a consistent system state from which all running applications can be safely resumed following a fault (*col. 1 lines 13-17*).

Chung et al. and Stiffer et al. does not explicitly teach an apparatus comprising of means to enables remote direct memory read write access to a persistent memory, wherein the persistent memory unit is addressable at byte-level granularity.

Pandya teaches the TCP/IP processor and engine using RDMA implementing a RDMA engine that provides various capabilities necessary for enabling remote direct memory access (*paragraph 0124 lines 13-15*) where the RDMA transfer may include read and write operation from a region with a certain number of bytes with a specific offset (*paragraph 0124 lines 80-86*).

Refer to claim 1 for motivational statement.

In regard to claim 27, Chung et al. does not explicitly teach the apparatus of claim 26, further comprising: means for allowing the primary process and the backup process to access context information regarding the addresses.

Stiffer et al. teach the method of checkpointing and fault recover software runs on standard platforms to detect observable malfunctions such as out-of-range address or exceeding a allocated range defined for a given data structure (*col. 4 lines 24-36*).

Refer to claim 26 for motivational statement.

In regard to claim 28, Chung et al. teach the apparatus of claim 26, further comprising: means for allowing the backup process to access the checkpoint data upon failure of the primary process (*Checkpoint server transmit last stored state to new primary application module, fig. 1*,

110, H1-6, col. 4 lines 41-48).

10. Claim 24 is rejected under 35 U.S.C. 103(a) as being unpatentable over Chung et al. (US 6,195,760) in further view of Stiffer et al. (US 6,622,263) in further view of Pandya (US 2004/0037319) in further view of Wang (US 7,082,553).

In regard to claim 24, Chung et al. and Stiffer et al. Pandya does not explicitly teach the computer product of claim 21, further comprising: computer executable instructions operable to: allow the backup process to access the multiple sets of checkpoint data at one time.

Wang teaches the for providing reliability and availability in a distributed component object model object oriented system by implementing checkpoint request to store current checkpoint of each object in registry image of disk storage (*fig. 8, col. 9 lines 21-30*).

It would have been obvious to modify the method of Chung et al. and Stiffer et al. and Pandya by adding Wang distributed component object model object oriented system. A person of ordinary skill in the art at the time of applicant's invention would have been motivated to make the modification because it would help to quickly recover from a failure (*col. 9 lines 30-46*).

11. Claims 29-31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chung et al. (US 6,195,760) and in further view of Stiffer et al. (US 6,622,263) in further view of Pandya (US 2004/0037319) in further view of St. Pierre et al. (US 6,141,773).

In regard to claim 29, Chung et al. teach the apparatus of claim 26, further comprising:
means for overwriting the checkpoint data with current checkpoint data (*when a failure of the primary application, the checkpoint data of the last stored state of the failed primary is supplied to the backup application module, col. 1 lines 49-57*).

Chung et al. and Stiffler et al. and Pandya does not explicitly teach the means for creating multiple sets of checkpoint data by appending updated checkpoint data to at least one previous set of the checkpoint data;

St. Pierre et al. disclosed the method of backing up and restoring data in a computer storage system where differential backup is formed by the identified changed segments omitting at least on the segments that has not been changed (*col. 5 lines 30-63*). A differential bit file may be constructed including copies of only the changed data segments (*fig. 13, 111a-111d*). The differential bit file captures changes to a logical entity as contiguous (*col. 17 lines 28-38*).

Refer to claim 14 for motivational statement.

In regard to claim 30, Chung et al. disclosed the apparatus of claim 29, further comprising: means for periodically supplying (*Checkpoint server periodically receives from each fault-protected application module running on the network, fig. 1, col. 4 lines 41-44*) at least a

portion of the multiple sets of checkpoint data (*snapshot of the running state of the primary application, col. 1 lines 49-58*) in the backup process (*Replica-Manager stores information necessary to effect recovery of an entire host computer running several different application modules, fig. 2, 200, col. 5 lines 21-30*).

In regard to claim 31, Chung et al. disclosed the apparatus of claim 30, further comprising: means for providing previously unread portions of the checkpoint data to the backup process upon failure of the primary process (*upon failure detection of an application module, the last stored state of the failed application is retrieved form the memory of Checkpoint Server, col. 4 lines 45-48*).

Conclusion

12. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. See PTO 892.

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event,

Art Unit: 2114

however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to LOAN TRUONG whose telephone number is (571) 272-2572. The examiner can normally be reached on M-F from 8am-4pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, SCOTT BADERMAN can be reached on (571) 272-3644. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Loan Truong
Patent Examiner
Art Unit: 2114



SCOTT BADERMAN
SUPERVISORY PATENT EXAMINER